Amendments to the Claims:

This listing of claims replaces all prior versions and listings of claims in the application:

Listing of Claims:

1. (Currently amended) A computer program product, tangibly embodied in an information carrier, the computer program product being operable to cause data processing apparatus to ~~instruction comprises~~ :

~~a branch instruction that causes~~ cause an executing instruction stream to branch to an instruction at an address specified in the instruction if a state of a specified state name is a specified value.

2. (Currently amended) The ~~instruction~~ computer program product of claim 1 wherein the state is set to a logic one or a logic zero by a processor and indicates the currently processing state.

3. (Currently amended) The instruction computer program product of claim 1 wherein the state is set to logic one or a logic zero by a microengine in a parallel processor and indicates ~~the~~ a currently processing state.

4. (Currently amended) The instruction computer program product of claim 2 wherein the state is available to ~~all~~ a plurality of microengines.

5. (Currently amended) The ~~instruction~~ computer program product of claim 1 further comprising:

an optional token that is set by a programmer and specifies a number i of instructions to execute following the branch ~~instruction~~ before performing the branch operation where the number of instructions can be specified as one, two or three.

6. (Currently amended) The ~~instruction~~ computer program product of claim 1 wherein causing the executing instruction stream to branch ~~the instruction~~ has the following format:
br_inp_state[state_name, label#], optional_token.

7. (Currently amended) The ~~instruction~~ computer program product of claim 1 wherein causing the executing instruction stream to branch ~~the instruction~~ causes a branch if the value of the specified state name is set to a logic one.

8. (Currently amended) The ~~instruction~~ computer program product of claim 1 wherein ~~the instruction~~ causing the executing instruction stream to branch causes a branch if the value of the specified state name is cleared to a logic zero.

9. (Currently amended) The ~~instruction~~ computer program product of claim 1 wherein the state name is the name assigned to an executing context.

10. (Original) A method of operating a processor comprises:

evaluating a value of a specified state name; and

performing a branching operation based on the value of the specified state name being set or cleared.

11. (Original) The method of claim 10 wherein the state is set to a logic one or a logic zero by a processor and indicates the currently processing state.

12. (Original) The method of claim 11 wherein the state is set to logic one or a logic zero by a microengine in a parallel processor and indicates the currently processing state.

13. (Original) The method of claim 11 further comprising:

branching to an instruction at a branch target field specified as a label in the instruction.

14. (Original) The method of claim 11 further comprising:

executing a number i of instructions following execution of the branch instruction before performing the branch operation based on evaluating an optional token that is set by a programmer.

15. (Original) The method of claim 11 further comprising:

evaluating an optional token that is set by a programmer; and

executing one instruction following execution of the branch instruction before performing the branch operation based on whether the optional token is set.

16. (Original) A processor comprises:

a register stack;

an arithmetic logic unit coupled to the register stack and a program control store that stores a branch instruction that causes the processor to:

evaluate a value of a specified state name; and

perform a branching operation based on the value of the specified state name being set or cleared.

17. (Original) The processor of claim 16 wherein the processor is a parallel processor with the register stack an arithmetic logic unit is part of a microengine, and the processor further comprises:

at least an additional microengine, the microengine comprising:

a register stack;

an arithmetic logic unit coupled to the register stack and a program control store, and wherein the state is set to logic one or a logic zero by one of the microengines and indicates the currently processing state.

18. (Original) The processor of claim 16 further comprising:

a branch target field specified as a label in the instruction.

19. (Original) A computer program product residing on a computer readable medium for causing a processor to perform a function comprises instructions causing the processor to:

evaluate a value of a specified state name; and

perform a branching operation based on the value of the specified state name being set or cleared.

20. (Original) The computer program product of claim 19 wherein instructions to perform a branching operation branch if the value of the specified state name is set to a logic one.